

## PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-154078

(43)Date of publication of application : 09.06.1998

(51)Int.Cl.

G06F 9/445

G06F 13/00

(21)Application number : 09-265696

(71)Applicant : INTERNATL BUSINESS MACH  
CORP <IBM>

(22)Date of filing : 30.09.1997

(72)Inventor : MARCY DEBARAKONDA  
AJAI MOOINDORA  
DEBORAH JEAN ZHUKOVSKI

(30)Priority

Priority number : 96 722434

Priority date : 07.10.1996

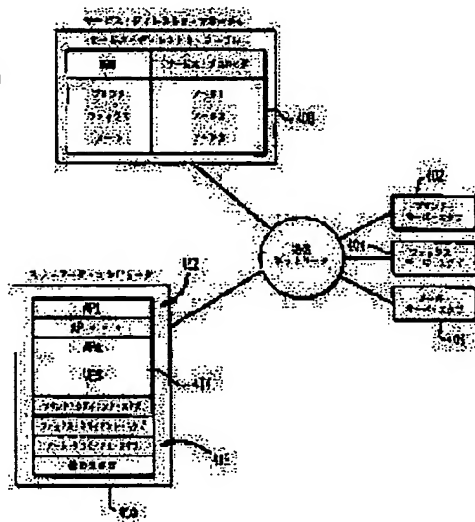
Priority country : US

## (54) ACCESS SUPPLY METHOD FOR NETWORK SERVICE

## (57)Abstract:

**PROBLEM TO BE SOLVED:** To realize the service of a network base by detecting requested remote network service and down-loading an object into the memory of a network node.

**SOLUTION:** A service directory table(SDT) 408 contains information on three service providers. Various applications 412, the action virtual environment supervisor(VES) object 414 containing the table of constituted service and a passive stub object 416 supplying connection to service are contained in the specified instance of a network computer 410. When the application requests access to remote network service, the remote service network is detected, the object is down-loaded into the memory of the network node and the application executes access when remote network service is requested.



BEST AVAILABLE COPY

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平10-154078

(43) 公開日 平成10年(1998) 6月9日

(51) Int.Cl.<sup>8</sup>G 0 6 F 9/445  
13/00

識別記号

3 5 7

F I

G 0 6 F 9/06  
13/004 2 0 J  
3 5 7 Z

審査請求 未請求 請求項の数20 O L (全 13 頁)

(21) 出願番号 特願平9-265696

(22) 出願日 平成9年(1997) 9月30日

(31) 優先権主張番号 0 8 / 7 2 2 4 3 4

(32) 優先日 1996年10月7日

(33) 優先権主張国 米国 (US)

(71) 出願人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション

INTERNATIONAL BUSINESS MACHINES CORPORATION

アメリカ合衆国10504、ニューヨーク州  
アーモンク (番地なし)

(72) 発明者 マーシー・デバラコンダ

アメリカ合衆国10510、ニューヨーク州ブ  
ライアークリフ・マナー、キウエイ・カ  
ープ 611

(74) 代理人 弁理士 坂口 博 (外1名)

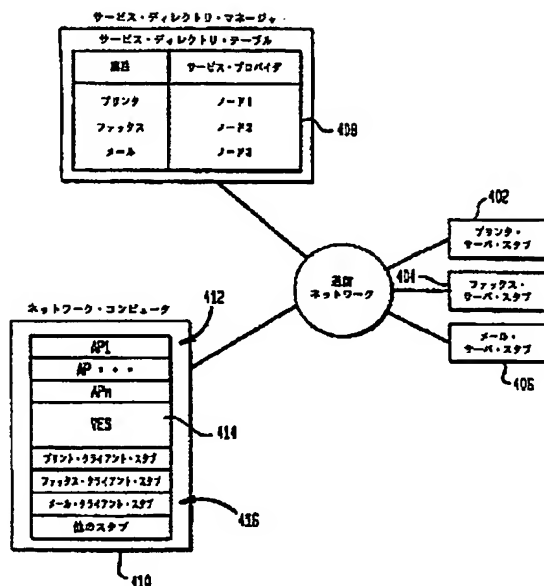
最終頁に続く

(54) 【発明の名称】 ネットワーク・サービスへのアクセス提供方法

(57) 【要約】

【課題】 クライアントのために、ネットワーク・ベースのサービスを可能にする、柔軟で強力且つ移植性のある手段を提供する。

【解決手段】 ネットワーク・コンピュータ上で実行されるアプリケーションに、ネットワーク・サービスへのアクセスを提供する、VEMと呼ばれるダウンロード可能な即時ミドルウェアのためのシステム及び方法が提供される。ネットワーク・サービスには、印刷及びローカル記憶などのシステム・サービスが含まれる。VEMがデフォルト指定のクライアント・サービスを構成し、これらのサービスに関する情報を記憶する。ネットワーク・コンピュータ上で実行されるアプリケーションがサービスの1つを使用したいとき、これはそのローカルVEMと通信する。VEMはハンドルを適切なサービスに返却し、サービス要求を完了する。



**【特許請求の範囲】**

【請求項 1】 ネットワーク・クライアント上で実行されるアプリケーションに、ネットワーク・サービスへのアクセスを提供する方法であって、

前記アプリケーションがリモート・ネットワーク・サービスへのアクセスを要求する時を判断するステップと、前記アプリケーションが前記リモート・ネットワーク・サービスへのアクセスを要求したとの判断にตอบสนองして、前記アプリケーションにより要求された前記リモート・ネットワーク・サービスを突き止めるステップと、少なくとも 1 つのオブジェクトをネットワーク・ノードのメモリにダウンロードするステップと、を含み、前記アプリケーションが前記リモート・ネットワーク・サービスを要求時にアクセスすることを可能にする方法。

【請求項 2】 前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・スタブをダウンロードするステップを含む、請求項 1 記載の方法。

【請求項 3】 前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・ハンドルをダウンロードするステップを含む、請求項 1 記載の方法。

【請求項 4】 前記ネットワーク・クライアントのユーザに、以前にアクセスが許可された使用可能なネットワーク・サービスのビジュアル表示を提供するステップを含む、請求項 1 記載の方法。

【請求項 5】 前記ネットワーク・クライアントがネットワーク・コンピュータである、請求項 1 記載の方法。

【請求項 6】 前記ネットワーク・ノードが J A V A 端末である、請求項 1 記載の方法。

【請求項 7】 前記ネットワーク・ノードがパーソナル・デジタル・アシスタント（P D A）である、請求項 1 記載の方法。

【請求項 8】 前記ネットワーク・サービスがシステム・サービスである、請求項 1 記載の方法。

【請求項 9】 前記システム・サービスが印刷サービスである、請求項 8 記載の方法。

【請求項 10】 前記システム・サービスが永久記憶である、請求項 8 記載の方法。

【請求項 11】 前記システム・サービスがシステム・モニタリングである、請求項 8 記載の方法。

【請求項 12】 前記システム・サービスがシステム管理である、請求項 8 記載の方法。

【請求項 13】 仮想環境マネージャが前記ネットワーク・サービスへのアクセスを提供する、請求項 1 記載の方法。

【請求項 14】 前記仮想環境マネージャが、前記ネットワーク・ノード上で実体化される仮想マシン上で実行される、請求項 13 記載の方法。

【請求項 15】 前記ネットワーク・サービスへのアクセスがもはや必要とされない時を判断し、前記ネットワーク・ノードの前記メモリから前記オブジェクトの除去を開始するステップを含む、請求項 1 記載の方法。

【請求項 16】 前記リモート・サービス・プロバイダにより、前記ネットワーク・クライアントが使用可能なサービスのセットを、第 3 者ディレクトリに登録するステップを含み、前記オブジェクトが前記ディレクトリを参照して、前記サービスへのアクセスを獲得する、請求項 1 記載の方法。

【請求項 17】 前記ネットワーク・ノード上で実行される複数のアプリケーションにより、前記サービスへのアクセスを共用するステップを含む、請求項 1 記載の方法。

【請求項 18】 あるアプリケーションにより必要とされるサービスへのアクセスを、前記アプリケーションにのみ知れる固有のキーによってのみ提供することにより、前記サービスを他のアプリケーションから分離するステップを含む、請求項 1 記載の方法。

【請求項 19】 ネットワーク・コンピュータ上で実行されるアプリケーションに、システム・サービスへのオンデマンド・アクセスを提供する方法であって、

a) ユーザからの要求にตอบสนองして、リモート・ネットワーク・サービスへのアクセスを提供する仮想環境マネージャをダウンロードするステップと、

b) システム・サービスのセットを識別するステップと、

c) 適切なリモート・サービス・プロバイダに連絡し、各前記サービスに対するクライアント・スタブまたはハンドルをダウンロードすることにより、前記システム・サービスのセットを構成するステップと、

d) 構成済みサービスのリストを保持するステップと、

e) 前記構成済みサービスへのアクセスを提供するステップと、

を含む、方法。

【請求項 20】 複数のアプリケーションにより、前記システム・サービスへのアクセスを共用するステップを含む、請求項 19 記載の方法。

**【発明の詳細な説明】****【0001】**

【発明の属する技術分野】 本発明はネットワーク・コンピュータに関する。より詳細には、本発明は、ネットワーク・サービス（例えば印刷及びローカル記憶などのシステム・サービス）へのアクセスを、ネットワークに接続されるコンピュータ上で実行されるアプリケーションに提供する方法及び手段に関する。

**【0002】**

【従来の技術】 ネットワーク・コンピュータのパラダイムが普及すると、ネットワーク・コンピュータとして知られる、独立の存在のための手段を有さない単純で安価

なデスクトップ・コンピュータが一般的となる。こうした装置は(単純化のために)、低機能のオペレーティング・システムを実行し、ローカル記憶、印刷及びモニタリングなどの基本システム・サービスをサーバに頼り得る。PCなどの従来のクライアントは、それら自身の基本システム・サービスを提供する一方で、ネットワークからも追加のサービスを求めがちである。

【0003】これらの基本システム・サービスをアプリケーションに提供するために使用される既存のアプローチは、ネットワーク・コンピュータ上に完全機能の動作環境を存在させたり、或いは各アプリケーションが必要なサービスの自身のセットを提供することを要求した。前者のアプローチはネットワーク・クライアントにとって実現可能でない。なぜなら、これらのコンピュータは(例えばネットワーク・コンピュータの場合、)完全機能の動作環境をサポートするための、必ずしも十分な物理資源を装備していないからである。こうした物理資源には、物理メモリ及び接続周辺装置(例えばディスク・ドライブ)などが含まれる。後者のアプローチでは、各アプリケーションが必要な基本システム・サービスのサポートを提供できるように、それが実行されるプラットフォーム及び環境を意識しなければならない欠点を有する。更に前者のアプローチは、ネットワーク・クライアントのコストを追加し、後者のアプローチは、アプリケーションの設計の複雑度を増す。

【0004】

【発明が解決しようとする課題】本発明の目的は、クライアントのために、ネットワーク・ベースのサービスを可能にする、柔軟で強力且つ移植性のある手段を提供することである。

【0005】

【課題を解決するための手段】好適な実施例では、仮想環境マネージャ(VEM)と呼ばれるダウンロード可能なミドルウェアが提供される。VEMはアプリケーションがクライアント・コンピュータ及びそれが接続されるサーバのアーキテクチャ及び環境に完全に独立に開発されることを可能にする。クライアント(すなわちネットワーク・コンピュータ)がサービスをアクセスするために、VEMは1つまたは複数の接続サーバ上で使用可能な、サービス・ディレクトリ・テーブルを照会する。サービス・ディレクトリ・テーブルへのアクセスは、指示されたサービス・プロバイダに接続するためのハンドルを返却する。

【0006】

【発明の実施の形態】本発明と一緒に使用するのに好適な疎結合のシステムが、図1に示される。システムは、通信ネットワーク112により相互接続される複数のコンピュータ102乃至106を含む。これらのコンピュータの幾つかは、ネットワーク・コンピュータ106(1)、106(2)として知られる。なぜならこれら

は、それらの基本機能の多くを提供するためにネットワークを介して使用可能なサービスに頼るからである。他のコンピュータは、サービス・プロバイダ104

(1)、104(2)、104(3)として知られる。なぜなら、これらは基本システム・サービスなどのネットワーク・サービスをネットワーク・コンピュータに提供するからである。コンピュータの幾つかは、サービス・ディレクトリ・マネージャ(SDM)102としても知られる。なぜならこれらは、サービス・プロバイダによりネットワーク・コンピュータに提供されるサービスのリストを保持するからである。

【0007】ネットワーク・コンピュータ106が、例えばJAVA端末、パーソナル・デジタル・アシスタント(PDA)、またはインターネット端末上に実現され得る。通信プロトコルは、HTTP及びTCP/IPである。通信ネットワーク112は、例えばトークン・リングである。サービス・システム102、104

(1)、104(2)、104(3)は、例えばAIX 4.2を用いるIBM RISC System/6000マシン上で実現され得る。

【0008】論理構成が上述の物理システム上に配置され得る。この構成は、3つの階層のクライアント/サーバ機構により説明される。この機構では、階層1はクライアント機能を、階層2はサービス・プロバイダを、階層3はデータ・オブジェクト・サーバ(すなわち情報記憶装置)を表す。本実施例で述べられるシステムは、階層1及び階層2、並びにそれらの間のインタフェースに関連する。階層2と階層3との間のインタフェースは制限されず、従ってユーザは、従来の信頼の置けるデータ情報アクセス・システムに頼ることができる。サービス・プロバイダは、完全に階層2サーバ上に存在するか、或いは特殊なサービス・スタブを提供することにより、その機能を階層1及び階層2に割当てるように設計され得る。

【0009】各ネットワーク・コンピュータは、仮想環境マネージャ(VEM)108(1)、108(2)を含み、これらは通信ネットワーク112を通じてサービス・プロバイダからダウンロードされる。VEMはネットワーク・コンピュータのランダム・アクセス・メモリ(図示せず)内に実体化されるプログラム・コードとして実現される。特にVEMは(“オブジェクト指向プログラミング”の意味において、)VEMCLASS(表1で定義される)と呼ばれる基本クラスを提供し、これに頼って、オブジェクトの集合を定義する。各VEMに含まれるオブジェクトの1つは、仮想環境スーパーバイザ(VES)109(1)、109(2)と呼ばれる活動エンティティであり、これはVEM状態を保持する役割をする。好適な実施例では、VESは活動JAVAオブジェクトであり、VEMCLASSから直接実体化される。VEM状態は、構成済みサービスのテーブル110(1)、11

0(2)、及び活動アプリケーションのテーブル112(1)、112(2)を含む。構成済みサービスのテーブル及び活動アプリケーションのテーブルの両者も、各ネットワーク・コンピュータのランダム・アクセス・メモリ内で実体化される。アプリケーションはVESと対話するためにVEMCLASSから継承する。オブジェクト・クラスはアプリケーションにより編成され、アプリケーションのネーム空間に結合される。

【0010】本発明の実施例によれば、ネットワーク・コンピュータがスイッチ・オンされるとき、これはブート・プロセスを実行し、自身を使用可能な状態に準備する。ブート・プロセスの終了時にネットワーク・コンピュータがユーザに識別を催促する。識別プロセスは、例えばユーザ名及びパスワードをタイプ入力するなどである。ユーザ識別の検証の後、ネットワーク・コンピュータはユーザにデスクトップ環境を提供する。

【0011】デスクトップ環境は、好適にはJAVAコンテナである。JAVAコンテナはJAVA仮想マシン上で実行される。コンテナ及びJAVA仮想マシンの両者は、ネットワーク・コンピュータ上で実行されるウェブ・ブラウザ114(1)、114(2)(例えばネットスケープ・ナビゲータ)などのプログラムにより提供される。ブラウザはネットワーク・コンピュータのブート・シーケンスの間にダウンロードされるか、或いはネットワーク・コンピュータの統合部分として提供され得る。

【0012】初期化プロセスの一部として、ブラウザはユーザのためにVES及び構成ファイルをサービス・プロバイダからダウンロードする。VESは構成ファイルを用いて、ユーザのデスクトップ環境(デスクトップ)をセットアップする。デスクトップの例(ホームページの形式を取り得る)が図6に示される。構成ファイルは、システム及びユーザが最も頻繁にアクセスするアプリケーション・サービスのリストを含む。これらのシステム及びアプリケーション・サービスは、デスクトップ上に表示されるアイコン602またはボタン604、及び(または)テキスト・リスト606として実現され得る。システム及びアプリケーション・サービスはハイパリンクとして、または適切なJAVAコードがダウンロード済みの直接制御ボタンとしてアクセスされ得る。VEMの残りは、ユーザ・アプリケーションと一緒にダウンロードされる。

【0013】ホーム・ページの初期ロギング及びダウンロードが図2に示される。ステップ202で、ネットワーク・コンピュータがユーザにより、パワーオンまたはリセットされる。それに応答してステップ204で、ネットワーク・コンピュータが、そのマシン特有の従来のブート・シーケンスを開始する。このシーケンスは、ウェブ・ブラウザのダウンロードを含むように拡張され得る。ステップ206で、ネットワーク・コンピュータが

ユーザにユーザ名及びパスワードを催促する。ネットワーク・コンピュータはステップ208で、従来方法によりユーザ名及びパスワードが有効か否かを判断する。当業者には理解されるように、ステップ206及び208は、代わりにステップ204の一部としても実行され得る。ステップ208で、ユーザ名またはパスワードのいずれかが有効でないと判断されると、本方法はステップ206に戻る。ユーザ名及びパスワードが有効な場合には、ステップ210で、ネットワーク・コンピュータがVES及び構成ファイルをダウンロードする。

【0014】VESがユーザのためにダウンロードされると、これは構成ファイル内に存在する全てのサービスの構成を開始する。構成ファイルは従来のフラット・ファイル形式でよい。例えば構成ファイルは、ネットスケープ・ナビゲータ3.0により使用されるタイプのブックマーク・ファイルの形式である。この構成は、適切なサービス・プロバイダに連絡して、クライアントのサービス・テーブル110をサーバ情報により初期化することにより達成される。スタブもまた、階層1及び2上に存在するサービスのためにダウンロードされる。

【0015】現構成済みサービスに加え、VESはネットワーク上で使用可能な全てのサービスへのアクセスを有する。このことは、ユーザが任意の使用可能なサービスをデスクトップに追加し、結果的に、ユーザの構成ファイルに追加することを可能にする。

【0016】サービス・アクセスが、サービス・ディレクトリ・マネージャ(SDM)により提供される。SDMは、サービス・ディレクトリ・テーブル(SDT)

(図1参照)として参照されるサービスをテーブルを保持する。サービス・ディレクトリ・テーブルは、システム及びネットワーク上のサービス・プロバイダにより提供されるアプリケーション・サービスに関する情報を含む。ネットワーク上に複数のSDMが存在し得り、各々がSDTを保持するか、SDTの1つのインスタンスをアクセスする。

【0017】サービス・プロバイダがネットワークに接続されるとき、これは自身がSDMに提供するサービスのセットを公表する。これらのサービスのセットを公表するにおいて、各サービス・プロバイダにより実行されるステップが、図3に示される。ステップ302で、サービス・プロバイダは自身が提供するサービスのリストを生成する。ステップ304で、サービス・プロバイダはネットワーク内のサービスのリストを保持する責任を負うSDMを識別する。ステップ306で、サービス・プロバイダはサービスのリストを走査し、次に登録されるべきサービスを決定する。登録されるべき別のサービスが存在する場合、ステップ308で、サービス・プロバイダは登録メッセージをSDM(ディレクトリM)に送信し、次にステップ306に戻る。リスト内にもはやサービスが存在しない場合には、ステップ310で初期

化が完了する。ここでステップ306及び308は、1つのステップにより置換され得ることが理解されよう。その場合、リスト全体が1度だけ読出され、次に適切なSDMに1つのメッセージまたはメッセージ・シーケンスの一部として送信される。

【0018】様々なクライアント・スタブがネットワーク・コンピュータにおいて配置されるシステム状態の例が、図4に示される。図4は3つのサービス・プロバイダ、すなわちプリント・サーバ402、ファックス・サーバ404、及びメール・サーバ406を示す。SDT408は、3つのサービス・プロバイダに関する情報を含む。特に、情報はサービスのタイプ（属性）、及び特定のサービスを提供するサービス・プロバイダの位置を示す。ネットワーク・コンピュータ410の特定のインスタンスには、様々なアプリケーション412（AP1-APn）、構成済みサービスのテーブルを含む活動VESオブジェクト414、及びサービスへの接続を提供する受動スタブ・オブジェクト416（プリント・クライアント・スタブ、ファックス・クライアント・スタブ、メール・クライアント・スタブ）が含まれる。VEMはVES、スタブ、及び各アプリケーションのVEMCLASS部分を含む。

【0019】次にVEMについて詳述することにする。参考として、VEMのコード定義は、VEMCLASS、並びにクライアント及びサーバ・インタフェースを含む。これらは表1及び表2にそれぞれ示される。図5は、アプリケーションがネットワーク・サービスを使用したいときのフロー制御プロセスを示す。前述のように、クライアントへのVESのロードはブラウザにおいてHTMLページを見ることに類似である。VESがダウンロードされ、そのinit()メソッドが呼び出される。init()メソッドは、クラス変数VE\_supervisorが一度だけ初期化されるように同期する。VESを開始するためのあらゆる続く試みが不能にされる。VESはそれ自身をVEM\_idインスタンス変数をセットすることにより、id=0に割当てて。クラス変数num\_AFEが次に増分される。レジストリ（構成済みサービス・テーブル110及びアプリケーションのテーブル112を含む）、及び共用サービス・テーブルが作成される。最後に、ディレクトリ・サービス・リモート・オブジェクトが、htmlファイルに渡されるDSサーバ・パラメータを用いて、実体化される。

【0020】

【表1】/\*VESが実体化し、アプリケーションが継承するVEM基本クラスを下記に示す。\*/

```
public class VEMCLASS extends Applet implements VE
MBaseInterface {
/*VE_supervisorは、VES及び全てのアプリケーション
が使用可能な単一の変数*/
private static VEMCLASS VE_supervisor = null;
```

```
/*num_AFEは、アプリケーションIDを生成するために
使用される単一の単調増加のカウント*/
private static int num_AFE = 0;
/*VEM_idは、アプリケーションのIDの記憶を提供す
る、アプリケーション当たりの変数*/
protected int VEM_id;
/*レジストリは、構成済みサービス・テーブル及びア
プリケーション状態の記憶を提供する。*/
private Hashtable registry;
/*shared_service変数は、共用サービスの迅速な探索機
構を提供する。*/
private Hashtable shared_services;
/*String primary,secondary DS_Server変数は、SDM
の位置を記憶する。*/
private String primary_DS_server, secondary_DS_ser
ver;
/*directory変数はSDMへの直接アクセスを提供す
る。*/
private ServDir directory;
public VEMCLASS() {
}
public void init() {
}
/*表2に記述されるVEMBaseInterfaceの実現*/
public Cookie registerAFE(String name)
throws VEMRegistryException {
}
public Cookie registerService(String name)
throws VEMRegistryException {
}
public Cookie registerService(String name,Strin
g [] attrs)
throws VEMRegistryException {
}
public Cookie registerService(String name,boolean
shared)
throws VEMRegistryException {
}
public Cookie registerService(String name,Strin
g [] attrs,boolean shared)
throws VEMRegistryException {
}
public ServiceHandle getServiceHandle(Cookie c)
throws VEMRegistryException {
}
public Object getServiceInstance(Cookie c)
throws VEMRegistryException {
}
public boolean revokeService(Cookie c)
throws VEMRegistryException {
```

```

}
/*VES 状態を直接管理する内部ルーチン*/
synchronized private void addToRegistry(Cookie c,
RegistryElement r)
throws VEMRegistryException {
}
synchronized private RegistryElement getFromRegistry(Cookie c) {
}
synchronized private String removeFromRegistry(Cookie c)
throws VEMRegistryException {
}
synchronized private Cookie shareService(int o, String name) {
}
synchronized private void removeIfShared(int o, String name) {}
}

【0021】
【表2】/*VEM基本インタフェース*/
public interface VEMBaseInterface {
public static final boolean SHARED = true;
/*VEMSupervisorに、常駐アプリケーション・フロント・
エンドを通知する。アプリケーションの任意のVEM
状態情報への、アプリケーション及びVEMSupervisor
だけのアクセスを許可するために、固有キーが返送され
る。*/
/*アプリケーション・フロント・エンド(AFE)をV
EMに登録する。*/
public Cookie registerAFE(String name)
throws VEMRegistryException;
/*VEMSupervisorに、AFEにより必要とされる要求サ
ービスを通知するメソッド。この呼び出しは、VEM状
態をAFEが動的にアクセス可能なサービスへのハンド
ルにより更新済みである。AFEがハンドルを用いて、
サービスとの通信をセットアップする。固有キーが返送
され、サービス・ハンドルへのAFE排他アクセスを可
能にする。*/
/*VEM状態を、名前"name"を有する任意のサービスの
ためのハンドルにより、セットアップする。*/
public Cookie registerService(String name)
throws VEMRegistryException;
/*VEM状態を、名前"name"を有し、属性"attrs"を含
むサービスのためのハンドルにより、セットアップす
る。*/
public Cookie registerService(String name, String
[] attrs)
throws VEMRegistryException;
/*共用が"SHARED"に等しい、すなわち真ならば、名前"n

```

```

ame"を有するサービスがVEMに知っているか否かを確認
する。知れていないならば、VEM状態を共用サービ
スによりセットアップし、所有者をこのAFEにセッ
トする。関連キーを返却する。*/
public Cookie registerService(String name, boolean
shared)
throws VEMRegistryException;
/*共用が"SHARED"に等しい、すなわち真ならば、名前"n
ame"及び属性"attrs"を有するサービスがVEMに知れ
ているか否かを確認する。知れていないならば、VEM
状態を共用サービスによりセットアップし、所有者を
このAFEにセットする。関連キーを返却する。*/
public Cookie registerService(String name, String
[] attrs, boolean shared)
throws VEMRegistryException;
/*AFEがVEMに知れるサービスへのアクセスを獲得
するメソッド*/
/*サービスとの接続をセットアップするために使用され
得る、そのサービスに関連付けられるハンドルを獲得す
る。*/
public ServiceHandle getService(Cookie c)
throws VEMRegistryException;
/*しばしば共用サービスとして使用されるサービス・ス
タブへのアクセスを獲得する。*/
public Object getServiceStub(Cookie c)
throws VEMRegistryException;
/*全てのVEM状態からサービスを除去するメソッド*/
/*キーにより識別されるサービスを除去する。サービス
が共用される場合、所有者AFEだけがその除去を許
可される。*/
public boolean revokeService(Cookie c)
throws VEMRegistryException;
}

/*SDMインタフェース*/
public interface VEM_ServDir extends java.rmi.Remote
{
/*サービスを探索するメソッド。"失敗"とマークされる
場合、サービスに対してハンドルが返却される。クライ
アントはそれをどのように処理するかを判断する。返却
されるハンドルは、{<ip_address>:<port #>:<flag>}で
あろう。フラグは"失敗"または"活動中"であろう。*/
/*サービス名に一致する最初のサービスに対するサービ
ス・ハンドルを返却する。*/
String lookupService(String serviceName)
throws java.rmi.RemoteException;
/*サービス名及び全ての属性に一致する最初のサービス
に対するサービス・ハンドルを返却する。*/
String lookupService(String serviceName, String []
serviceAttr)
throws java.rmi.RemoteException;

```

```

/*サービス名に一致し、提供されたサービス・ハンドルの
“後”に来る“次の”サービスに対するサービス・ハンド
ルを返却する。*/
String lookupService(String ServiceName, String Ser
viceHandle)
throws java.rmi.RemoteException;
/*サービス名及び全ての属性に一致し、提供されたサー
ビス・ハンドルの“後”に来る“次の”サービスに対するサ
ービス・ハンドルを返却する。*/
String lookupService(String ServiceName, String Ser
viceHandle, String [] ServiceAttr)
throws java.rmi.RemoteException;
/*サービス・カウントを獲得するメソッド。“失敗”とマ
ークされる場合にもカウントされる。*/
/*一致するサービス名に対するカウントを返却する。*/
int getServiceCount(String ServiceName)
throws java.rmi.RemoteException;
/*一致するサービス名及び属性に対するカウントを返却
する。*/
int getServiceCount(String ServiceName, String [] S
erviceAttr)
throws java.rmi.RemoteException;
/*サービスに関するメタ情報を返却するメソッド。メタ
情報は“活動中”サービスと同様、“失敗”サービスを含
む。*/
/*ストリングの配列を返却する。第1の要素はサービス
の名前であり、残りは属性である。入力はサービス・ハ
ンドルである。*/
String [] getServiceNameAttr(String ServiceHandle)
throws java.rmi.RemoteException;
/*ストリングの配列を返却する。その配列の各要素は、
特殊な区切り文字“$”により分離されるサービスの属性
を含む。ここでサービス名、属性及びサービス・ハンド
ルが、“$”を含まないように規定する。*/
String [] getServiceNameAttr(String ServiceName)
throws java.rmi.RemoteException;
/*ストリングの配列を返却する。その配列の各要素は、
最初にサービス名を、続いてサービス属性を含む。全て
は“$”文字により区切られる。*/
String [] getListOfServices()
throws java.rmi.RemoteException;
}

```

【0022】VEMがダウンロードされ、初期化されるとアプリケーション（AP）がダウンロードされる。  
（ここでの議論では、APがアプレットと仮定する。VEMはアプリケーションをサポートするが、アプレットは改良済みの管理能力に対して好適である。）そのinit()メソッドにおいて、APはregisterAFE()メソッドを呼び出すべきである。この呼び出しは、APのインスタンス変数VEM\_idをクラス変数num\_AFEにセットし、num\_A

FEを増分する。num\_AFEは決して減分されることがないので、APはこの時点で固有のidを有することになる。次に、レジストリ内にAPのエントリ（アプリケーションのテーブル112）を作成するために、VESが呼び出される。この時点でエントリはアプレット・オブジェクト参照、id及び名前だけを含むが、必要に応じて後に拡張され得る。registerAFE()メソッドは、レジストリ内のAP情報へのアクセスが制御されるように、“クッキー（cookie）”と呼ばれる固有キーをAPに返却する。この時点で、APがVEM環境に完全に統合され、VEM機能が実行可能になる。

【0023】APはサービスを必要とするとき、最初にregisterService()メソッドの1つを用いて、VEMにサービスを登録するように要求する。サービス登録に際してVESはlookupService()メソッドを用いて、ディレクトリ・サービス（SDM）をチェックする。このメソッドはサービスへのハンドルを獲得し、サービスがスタブを有するか否かを確認する。肯定の場合、スタブがダウンロードされる。VEMは次に、サービス・エントリをレジストリ（構成済みサービス・テーブル112）に追加する。レジストリはサービスのハンドル、及び適宜スタブの参照を含む。クッキーがAPに戻される。サービスがレジストリに追加されると、AP及びVESの両方が必要時にそれをアクセスできる。

【0024】呼び出されるregisterService()メソッドの選択は、1）総称サービスが必要とされるか否か（すなわち、名前だけで識別可能なサービス）、或いは更にカスタム・サービスが必要とされるか否か（例えば名前同様、属性のリストにより識別されるサービス）、及び2）サービスが共用されるか否かに依存する。サービスが共用されない場合、サービスを登録するAPはクッキーへのアクセスを獲得するサービスだけである。それ以外では、サービスが存在すればクッキーが戻される。これはサービスに対して他のAPに与えられるクッキーと同じである。共用されるサービスがまだVEM内で使用可能でない場合、shared\_service変数が、新たな共用サービスの可用性を反映するように更新され、新たなクッキーがAPに戻される。

【0025】共用サービスへのアクセスは、アクセス制御リストを用いて制御され得る。代わりに、共用サービスが大域的に使用可能である、すなわち任意のAPが単にサービスを求めることにより、そのサービスに対してクッキーを受信できてよい。共用サービスは、クライアント上に存在するサーバ・スタブの数を最小化するのに有用である。これらはまた、あるAPにより引き起こされるサーバ振舞いの変化が、別のAPにより相互的に使用される場合にも有用である。

【0026】共用サービスは様々な方法でVEMから除去され得る。第1の実施例によれば、基準カウントが各サービスに対して保持される。この基準は、現在幾つの

APがサービスへのアクセスを有するかを示す。基準カウントが0に達するとき、サーバがVESにより除去される。別の実施例によれば、最初にサービスを取り込んだAPだけが、それを除去することを許可される。

【0027】そのサービスを含むようにVEM状態が更新されると、APはgetServiceHandle()メソッドを発行することにより、ハンドルを獲得するか、或いはgetServiceInstance()メソッドを用いることにより、スタブを獲得できる。この時、APは自由に任意の種類の接続を形成し、サービスを使用することができる。VEMは、AP及びサービスが如何に通信するかを制限するように実現され得る。サービスがもはや必要とされないとき、APはRevokeService()メソッドを呼び出し、これがVEM状態を次のように更新する。すなわち、サービスがAPに固有な場合、サービスが除去される。サービスが共用サービスの場合、それは上述の規則に従い除去される。

【0028】以上、本発明は好適な実施例により述べられてきたが、当業者には様々な変更及び改善が考案されよう。従って、ここで述べられた好適な実施例は範例として提供されたものであり、本発明を制限するものではないことを理解されたい。

【0029】まとめとして、本発明の構成に関して、以下の事項を開示する。

【0030】(1) ネットワーク・クライアント上で実行されるアプリケーションに、ネットワーク・サービスへのアクセスを提供する方法であって、前記アプリケーションがリモート・ネットワーク・サービスへのアクセスを要求する時を判断するステップと、前記アプリケーションが前記リモート・ネットワーク・サービスへのアクセスを要求したとの判断にตอบสนองして、前記アプリケーションにより要求された前記リモート・ネットワーク・サービスを突き止めるステップと、少なくとも1つのオブジェクトをネットワーク・ノードのメモリにダウンロードするステップと、を含み、前記アプリケーションが前記リモート・ネットワーク・サービスを要求時にアクセスすることを可能にする方法。

(2) 前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・スタブをダウンロードするステップを含む、前記(1)記載の方法。

(3) 前記ダウンロード・ステップが、適切なサービス・プロバイダに連絡し、前記リモート・ネットワーク・サービスに対するクライアント・ハンドルをダウンロードするステップを含む、前記(1)記載の方法。

(4) 前記ネットワーク・クライアントのユーザに、以前にアクセスが許可された使用可能なネットワーク・サービスのビジュアル表示を提供するステップを含む、前記(1)記載の方法。

(5) 前記ネットワーク・クライアントがネットワーク

・コンピュータである、前記(1)記載の方法。

(6) 前記ネットワーク・ノードがJAVAA端末である、前記(1)記載の方法。

(7) 前記ネットワーク・ノードがパーソナル・デジタル・アシスタント(PDA)である、前記(1)記載の方法。

(8) 前記ネットワーク・サービスがシステム・サービスである、前記(1)記載の方法。

(9) 前記システム・サービスが印刷サービスである、前記(8)記載の方法。

(10) 前記システム・サービスが永久記憶である、前記(8)記載の方法。

(11) 前記システム・サービスがシステム・モニタリングである、前記(8)記載の方法。

(12) 前記システム・サービスがシステム管理である、前記(8)記載の方法。

(13) 仮想環境マネージャが前記ネットワーク・サービスへのアクセスを提供する、前記(1)記載の方法。

(14) 前記仮想環境マネージャが、前記ネットワーク・ノード上で実体化される仮想マシン上で実行される、前記(13)記載の方法。

(15) 前記ネットワーク・サービスへのアクセスがもはや必要とされない時を判断し、前記ネットワーク・ノードの前記メモリから前記オブジェクトの除去を開始するステップを含む、前記(1)記載の方法。

(16) 前記リモート・サービス・プロバイダにより、前記ネットワーク・クライアントが使用可能なサービスのセットを、第3者ディレクトリに登録するステップを含み、前記オブジェクトが前記ディレクトリを参照して、前記サービスへのアクセスを獲得する、前記(1)記載の方法。

(17) 前記ネットワーク・ノード上で実行される複数のアプリケーションにより、前記サービスへのアクセスを共用するステップを含む、前記(1)記載の方法。

(18) あるアプリケーションにより必要とされるサービスへのアクセスを、前記アプリケーションにのみ知れる固有のキーによってのみ提供することにより、前記サービスを他のアプリケーションから分離するステップを含む、前記(1)記載の方法。

(19) ネットワーク・コンピュータ上で実行されるアプリケーションに、システム・サービスへのオンデマンド・アクセスを提供する方法であって、

a) ユーザからの要求にตอบสนองして、リモート・ネットワーク・サービスへのアクセスを提供する仮想環境マネージャをダウンロードするステップと、

b) システム・サービスのセットを識別するステップと、

c) 適切なリモート・サービス・プロバイダに連絡し、各前記サービスに対するクライアント・スタブまたはハンドルをダウンロードすることにより、前記システム・

サービスのセットを構成するステップと、  
d) 構成済みサービスのリストを保持するステップと、  
e) 前記構成済みサービスへのアクセスを提供するステップと、を含む、方法。

(20) 複数のアプリケーションにより、前記システム・サービスへのアクセスを共用するステップを含む、前記(19)記載の方法。

【図面の簡単な説明】

【図1】本発明と共に用いるのに好適な疎結合のシステムを示す図である。

【図2】ネットワーク・コンピュータのブート段階のフローチャートを示す図である。

【図3】各サービス・プロバイダにより実行される初期ステップのフローチャートを示す図である。

【図4】VEMが構成されたシステム状態の例を示す図である。

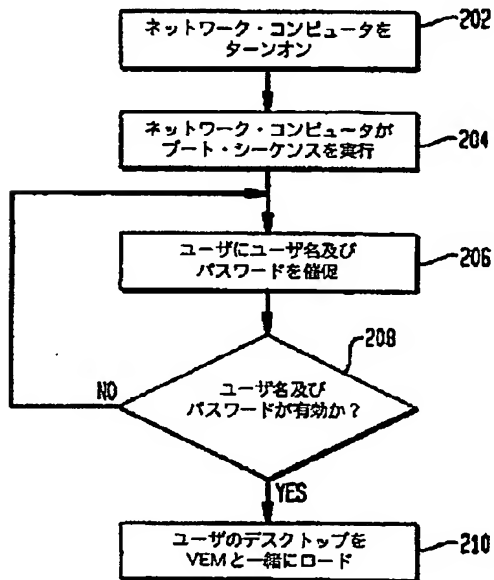
【図5】アプリケーションがネットワーク・サービスを使用しようとするときの、アプリケーションのフロー制御プロセスを示す図である。

【図6】デスクトップの例を示す図である。

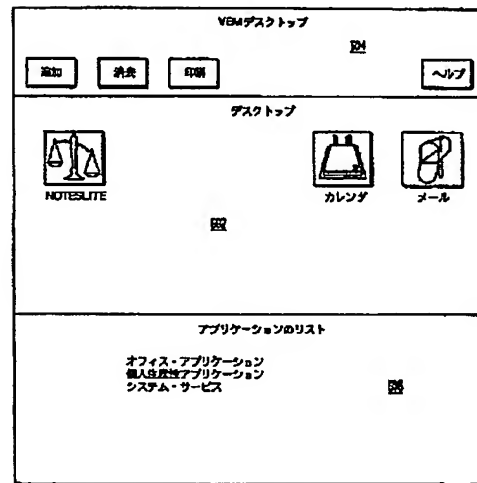
【符号の説明】

102 サービス・ディレクトリ・マネージャ (SDM)  
104 (1) (2) (3) サービス・プロバイダ  
106 (1) (2)、410 ネットワーク・コンピュータ  
108 (1) (2) 仮想環境マネージャ  
109 (1) (2) 仮想環境スーパーバイザ (VES)  
110 (1) (2) テーブル  
112 (1) (2) 通信ネットワーク  
114 (1) (2) ウェブ・ブラウザ  
402 プリント・サーバ  
404 ファックス・サーバ  
406 メール・サーバ  
408 SDT  
412 アプリケーション  
414 活動VESオブジェクト  
416 受動スタブ・オブジェクト  
602 アイコン  
604 ボタン  
606 テキスト・リスト

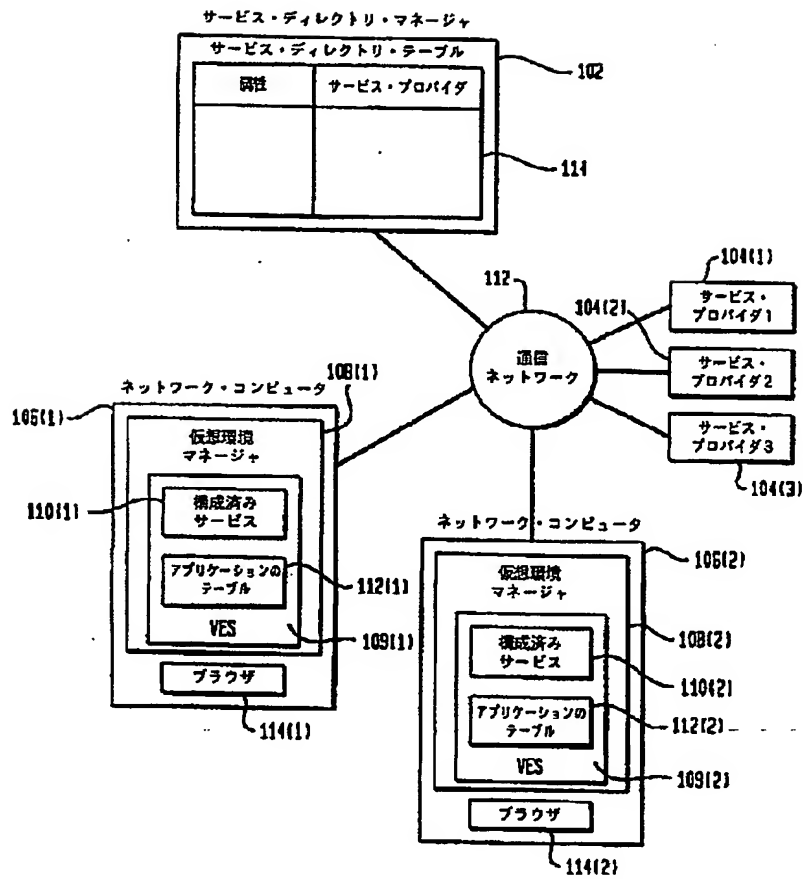
【図2】



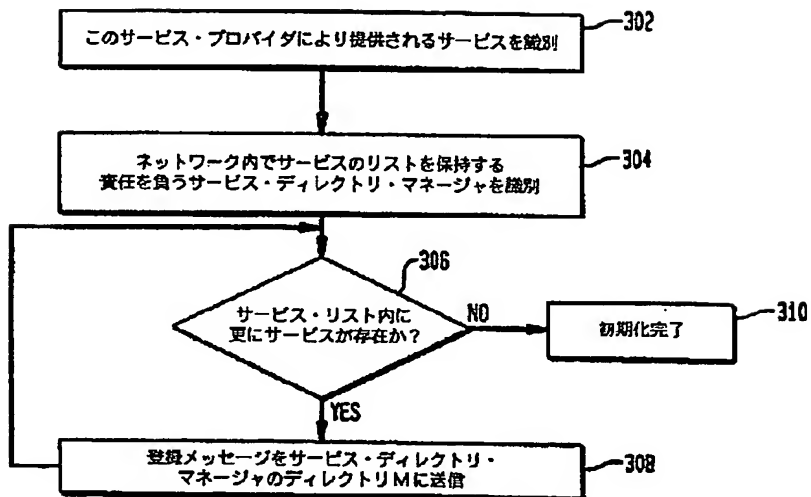
【図6】



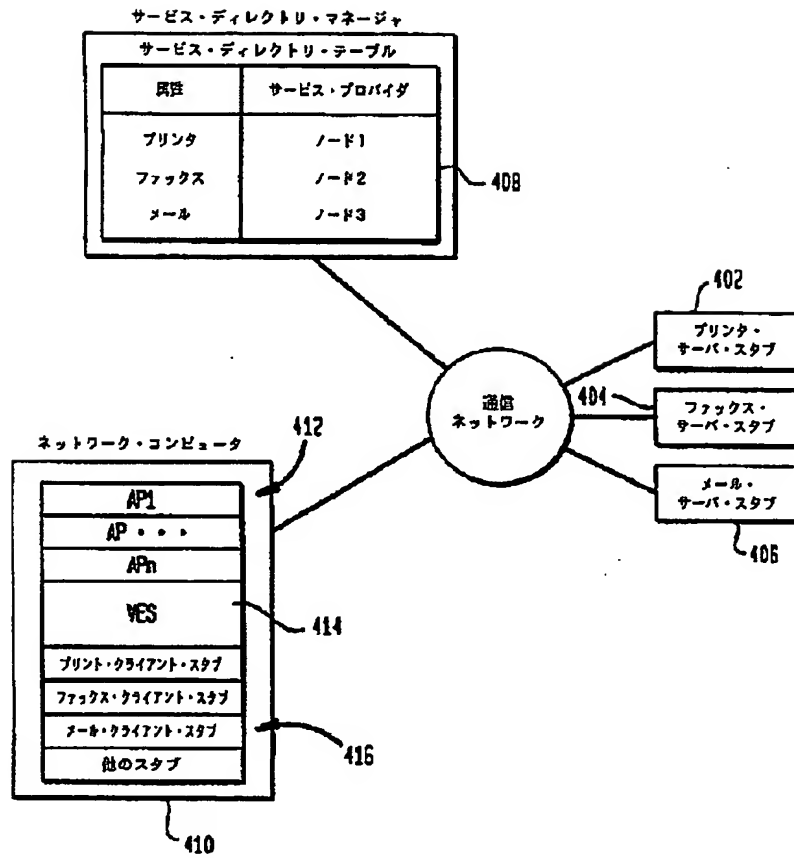
【図1】



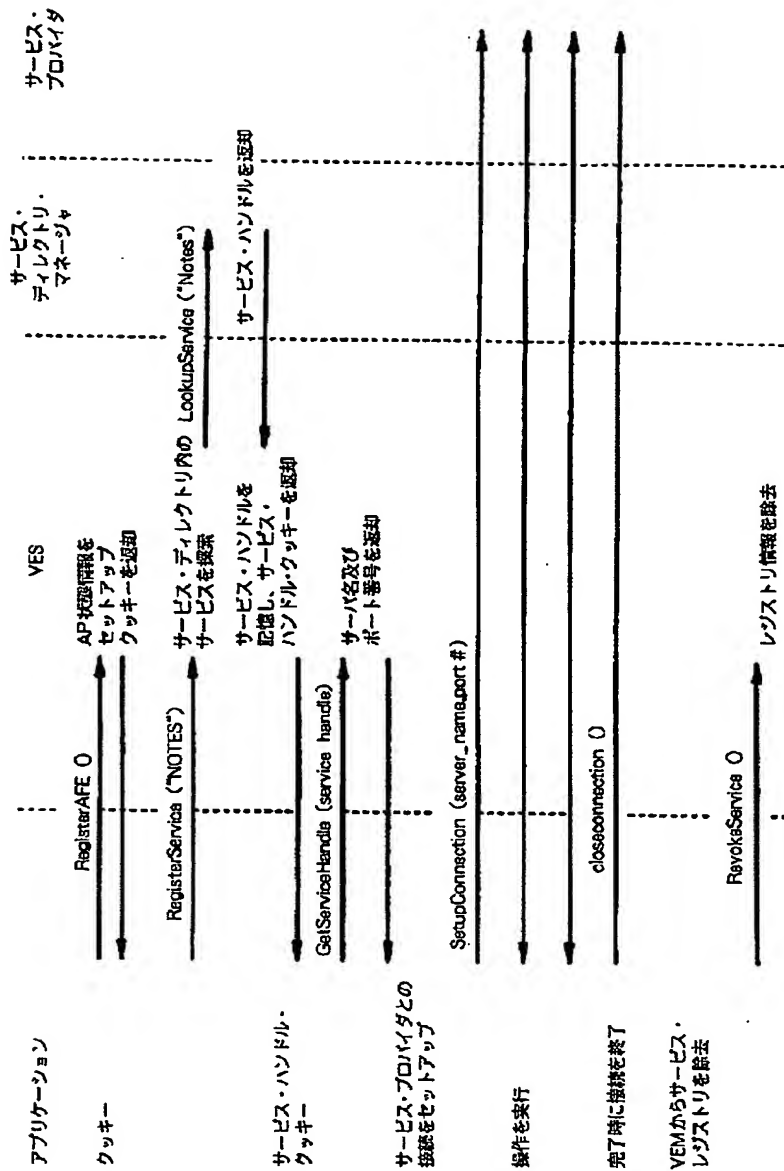
【図 3】



【図4】



【図5】



フロントページの続き

(72)発明者 アジャイ・モーインドラ  
アメリカ合衆国10598、ニューヨーク州ヨ  
ークタウン・ハイツ、リン・コート 1340

(72)発明者 デボラ・ジーン・ズコウスキー  
アメリカ合衆国10598、ニューヨーク州ヨ  
ークタウン・ハイツ、ビーバー・ドライブ  
3211

**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☒ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☒ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**